WEB-SCRAPING

STRUCTURE UNSTRUCTURED DATA

AGGREGATING WEB DATA SOURCE TO BUILD A SINGLE ONE

AUTHOR: NAMGYAL BRISSON

- INTRODUCTION
  - WHAT IS WEB-SCRAPING
  - LAW CONCERNS ABOUT WEB-SCRAPING
- XPATH, CSS & HTML QUICK OVERVIEW
  - HTML RENDERING EXAMPLE
  - QUERY BY XPATH SELECTOR
  - QUERY BY CSS SELECTOR
- IDENTIFY DATA TO SCRAPE
  - PREPARE QUERIES
- INTRODUCTION TO SCRAPY
  - CREATE THE SPIDER
- PREPARE DATA COLLECTION USING DJANGO
  - CREATE DJANGO MODELS
  - CONNECT THE SPIDER TO DJANGO
  - TRIGGER THE SPIDER TO FILL THE DB
- SCRAPYD INTRODUCTION
- TO GO FURTHER: DATA MANIPULATION & A.I
- QUESTIONS & ANSWERS

INTRODUCTION

- This presentation will introduce a full overview of Web-Scraping techniques.
- From data extraction to its representation.
- Scrapy, Scrapyd & Django will be used to achieve it.
- Full training material can be found on GitHub: https://github.com/nam4dev/web_scraping_presentation

WHAT IS WEB-SCRAPING

- ✓ Web-scraping is a technical approach which consists in extracting data from unstructured sources (websites, blogs, FTP, …) by downloading & parsing (HTML) pages.

- ✓ Detailed definition may be found at https://en.wikipedia.org/wiki/Web_scraping

LAW CONCERNS ABOUT WEB-SCRAPING

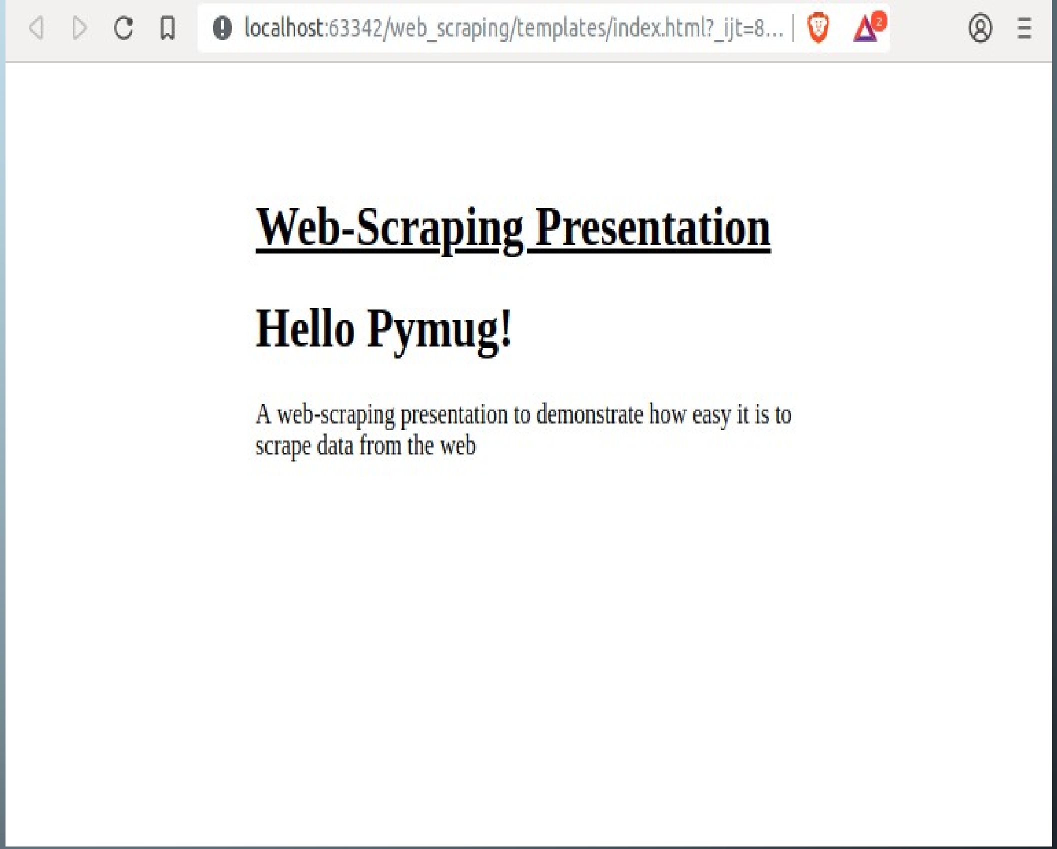
- **Conduct a Legal Review**
- Assess your project against following criteria:
  - Personal Data
  - Copyrighted Data
  - Database Data
  - Data Behind A Login
  - Sensitive Data
- https://en.wikipedia.org/wiki/Web_scraping#Legal_issues

XPATH, CSS & HTML QUICK OVERVIEW

- A web page is a Hyper Text Markup Language (HTML) tree
- It is composed of HTML nodes
- One can navigate the tree to reach content of interest

HTML RENDERING EXAMPLE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>[Web-Scraping Presentation] Hello Pymug</title>
</head>
<style>
    .container {
        width: 100%;
        position: relative;
    }
    .main {
        width: 600px;
        margin: 10% auto;
        position: relative;
    }
    .main h1.title {
        text-decoration: underline;
    }
</style>
<body>
    <div class="container">
        <div class="main">
            <h1 class="title">Web-Scraping Presentation</h1>
            <div>
                <h1>Hello Pymug!</h1>
                <p>
                    A web-scraping presentation
                    to demonstrate how easy it is
                    to scrape data from the web
                </p>
            </div>
        </div>
    </div>
</body>
</html>
```



Web-Scraping Presentation

Hello Pymug!

A web-scraping presentation to demonstrate how easy it is to scrape data from the web

QUERY BY XPATH SELECTOR

```
//h1[contains(“title”, @class)]/text()

=> “Web-Scraping Presentation”
```

QUERY BY CSS SELECTOR

h1.title ::text

=> “Web-Scraping Presentation”

IDENTIFY DATA TO SCRAPE

- GitHub Scrapy Official Repository
  - https://github.com/scrapy/scrapy/pulls
  - Data structure of a Pull Request Block:
    - Id **(need to be inferred from PR link)**
    - Title
    - Link
    - Status **(need to follow the PR link)**
    - Author **(need to follow the PR link)**
    - Scrapped URI (reference)

PREPARE QUERIES

Title:

xpath: //*[@data-hovercard-type="pull_request"]

→ css: a ::text

Link:

xpath: //*[@data-hovercard-type="pull_request"]

→ css: a::attr(href)

PREPARE QUERIES

Status:

xpath: //*[@id="partial-discussion-header"]/div[2]/div[1]/span

Author:

xpath: //*[@id="partial-discussion-header"]/div[2]/div[2]/a/text()

Pid:

Inferred from Link property

Scrapped URI:

Collected from the Response itself

INTRODUCTION TO SCRAPY

SCRAPY is a Python framework to scrape the web in an efficient & professional way.

Scraping service may periodically be triggered to fill a database to report concurrent prices on specific products, aggregate public data to build statistical views, etc.

Let’s get started!

CREATE THE SPIDER

https://scrapy.org/

Create the GitHub Spider which shall take in charge:

- GitHub scrapy pull requests page request
- Parsing data from the page
- Storing data into Django database

DATA COLLECTION USING DJANGO

Django is a famous python framework to build easy to highly complex website

We will take advantage of its “batteries included” to store and build quickly some views from scrapped data

CREATE DJANGO MODELS

What data do we need to store?

Pull Request Author:

- Name
- GitHub page link

Pull Request:

- Id
- Title
- Author (as foreign key)
- GitHub page link
- Status
- Scrapped URI

CONNECT THE SPIDER TO DJANGO

Connect the GitHub Spider to interact properly with Django:

- Create Spider’s items to represent data models to be inserted into db
- Implement into the Spider a Pipeline to insert scrapped item into db

TRIGGER THE SPIDER TO FILL THE DB

Trigger the spider manually to fill the database with scrapped data:

- Type in the command to start the spider
- Observe the logs
- Go to Django views to visualize scrapped data

SCRAPYD INTRODUCTION

https://scrapyd.readthedocs.io/en/stable/

Run the Scrapy daemon to trigger spider(s):

- Running scrapyd
- Using Scrapyd API through Django
- Trigger GitHub Spider
- To go further: full automation through celery

TO GO FURTHER: DATA MANIPULATION & A.I

Today, Artificial Intelligence lies on data, a big amount of data (big data) to train models accurately.

Scraping is a way to get that amount of data without much effort.

It is therefore a really good start to collect statistical data on which one could apply A.I algorithms

QUESTIONS & ANSWERS

BEING CLEVER IS TO FAIL AND LEARN FROM IT :)

THEREFORE, DO NOT HESITATE TO ASK ANY QUESTION

KNOWING THAT THERE'S NO STUPID QUESTION!!!